A Trace Based System for Decision Activities in CBM Process

Mohamed-Hedi KARRAY*, Brigitte CHEBEL-MORELLO*,
Noureddine ZERHOUNI*

**Femto-st Institute, Besancon 25000*
(Tel: +33 3 81 40 27 97; e-mail:hedi.karray@femto-st.fr,Brigitte.morello@femto-st.fr, noureddine.zerhouni@femto-st.fr).

Abstract: Prognostics and Health Management platforms are founded on Condition Based Maintenance process. Major works on this topic are focused on the diagnostic and prognostic modules and neglect the Decision Support Module which must be investigated to give more efficiency to PHM platforms. To improve this module with intelligence and rapidity we propose in this work to integrate a Trace Based System (TBS) into the decision support module. This TBS provides three main services (Traceability, self-learning and self-management) which are developed in this work.

Keywords: CBM, Intelligent Maintenance Process, Trace Based Systems, Knowledge Engineering.

1. INTRODUCTION

Maintenance has a strategic role in the industrial environment. Hence, while having this interest in company; industry has witnessed the development of different generations of maintenance systems. New technologies of information and communication (ICT) have helped to establish and evolve these systems. Thus, different maintenance strategies and concepts as PHM and e-maintenance are developed to improve efficiency of the maintenance process

The concept and framework of PHM have been developed based on well known maintenance methodologies and diagnostic techniques, such as preventative maintenance (PM), reliability centred maintenance (RCM) and condition based maintenance (CBM) (lee et al 2011).

In fact, to facilitate the CBM architectures development and to improve the interchangeability of different components, the MIMOSA consortium (Machinery Information Management Open System Alliance) proposed an Open System Architecture for Condition Based Maintenance (OSA-CBM) (Thurston 2001). OSA-CBM recommends the development of seven types of modules in order to cover the tasks that are useful to perform conditional and predictive maintenance:

- Sensor Module: provides system access to digitized sensor or transducer data.
- Signal Processing: performs signal transformations and CBM feature extraction.
- Condition Monitor: compares features against expected values and generates alerts.
- Diagnostic Processing: generates a diagnostic record (fault conditions and confidences).
- Prognostic Processing: projects the current health state into the future and estimates the Remaining Useful Life (RUL).
- Decision Reasoning (Reasoning Support Module): the primary function of the decision support module is to provide recommendations related to the maintenance action scheduled and also modifications of the equipment configuration or the mission profiles in order to accomplish the mission objectives. The decision support module needs to take into account the situation context that can be summarized by the operational history (including usage and maintenance), current and future mission profiles, high- level unit objectives, and resource constraints.
- Human Interface: provides an integrated user interface.

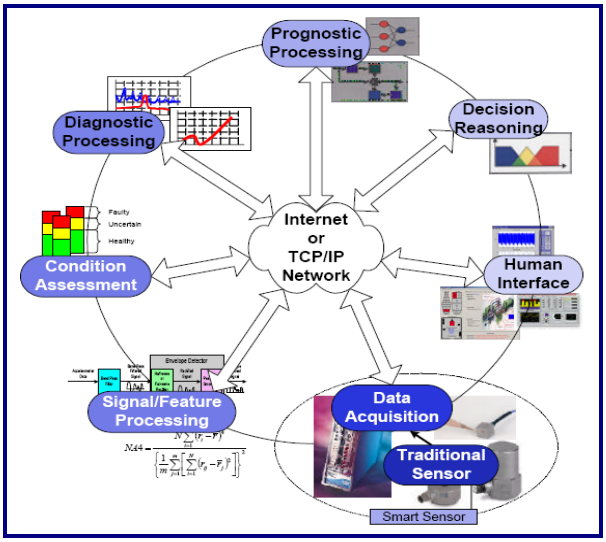

Fig. 1. CBM process' architecture.

These modules are generally connected via internet or a local network like in Fig.1.

In CBM process the output of the prognostic module is the RUL which is the input of the Decision Support Module. By some interactions between the users and the platform this module provides some recommendations in the scope of an operational scenario according to the context of the received RUL. Then the user of this platform (maintenance operator) takes the appropriate decision.

The recent works of PHM tend to focus on diagnostic and prognostic modules of this process. In order to give more efficiency to the CBM process The decision support module needs also to be developed and provides more t, smartness and a short time of reaction.

The integration of knowledge engineering into this module can give an added value by using its reasoning skills and knowledge reusing methods. As a variant of this engineering, we consider "Trace Engineering" is a good candidate to carry out the improvement of this module by adding back some auto-X features.

1.2 Trace Based Systems

Traces engineering concerns the management of traces for reuse in a system based on traces. Indeed, traces of interactions can be reused for two purposes namely assistance and analysis. A trace of interaction is defined as an object containing and representing a part of the experience of interaction between a system and its users. Also, (Champin et al., 2004) define the trace as a sequence of states and transitions representing user activities.

Treatments to be applied to traces from their collections are formalized using Trace Based System (TBS) (Cran et al 2007). Any system reusing traces is a TBS. This latter is based on three main phases: the collection often followed by a pretreatment step, analysis and exploitation (Bousbia 2011). To facilitate the establishment of a TBS, Settouti et al have proposed an architecture composed by several interconnected modules, as shown in Fig. 2 (Setoutti et al., 2006).

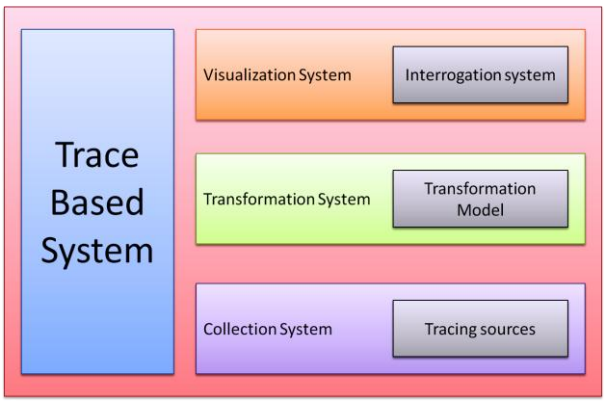

Fig. 2. TBS Architecture

The collection system captures the interactions through tracing sources, and creates a first trace. The system structures the collected traces in a hierarchical structure of classes called observed trace model (Cram et al 2007). The transformation system is the core of the TBS. It enables the generation of new knowledge from the collected traces. The choice of transformation model to be applied depends on the application of this trace. The set of traces collected and processed is then accessible through a query system and a visualization system to allow their exploit, analyze and interpret (Settouti et al., 2006).

Thus, integrates a TBS in a PHM platform including CBM process enables to involve the intelligence of the decision support module.

Hence, in this work we propose a Trace Based System for a PHM platform. This system is composed by three services that we called Traceability service, self-learning service and Self-management service. Each service is based on knowledge engineering methods and ensured automatically by the Decision Support Module (DSM) without any human intervention. This work will be based on a previous work that we made about self management in s-maintenance platform (Karray et al 2011).

The rest of this paper is organized as following: second section will be focused on the Trace Based System for PHM platform, the third, fourth and fifth sections would present respectively traceability, self-learning and Self-management services. In the sixth section we would present a use case of application of the proposed services. Finally we conclude the paper with some notes and future works.

2. Trace Based System for PHM

2.1 TBS adapted to PHM platform

Reusing traces in the context of PHM platform is up to snuff learning the decision-making tasks by the user of the platform during his interactions with it. Indeed, the reuse of interaction traces is an approach to the reuse of experience based on past experiences. Such approach is within the scope of the second generation expert system which is not based only on general knowledge but also has a new type of knowledge which is experience (Cram et al 2007).

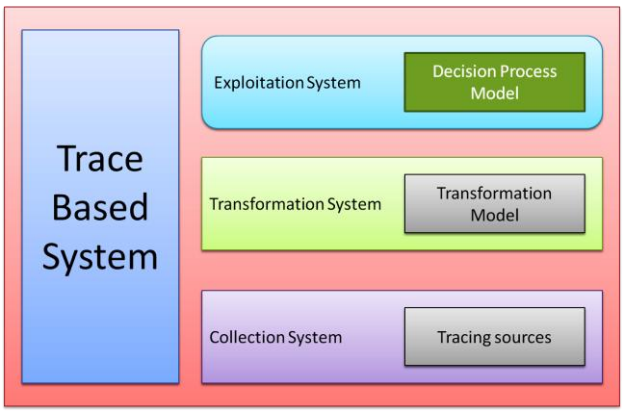

Fig. 3. TBS Architecture for PHM

The TBS aim in the Decisions Support Module is to ensure the interactions' self-learning that improves the decision-making tasks. Consequently these tasks can be self-managed

by the platform in the future executions without human intervention. So, analysis and exploitations will be provided by self-X services integrated into the module.

So, we changed the visualization system and interrogation by a generic exploitation that can be adapted to the purpose of traces' exploitation (see Fig.3). This system follows a process model that defines the process of exploitation. In our case, the process model will follow a maintenance processes model for self-manage processes after the execution of the learning service (traces analysis).

2.2 SBT process and functioning in the platform

We set up in Fig.4 an overview of a step by step closed-loop as part of the functioning of the envisaged TBS. Indeed, this TBS is based on a knowledge base including knowledge about maintenance domain as well as the knowledge model about maintenance processes. The steps in this loop are the inputs and outputs of the modules in the system.

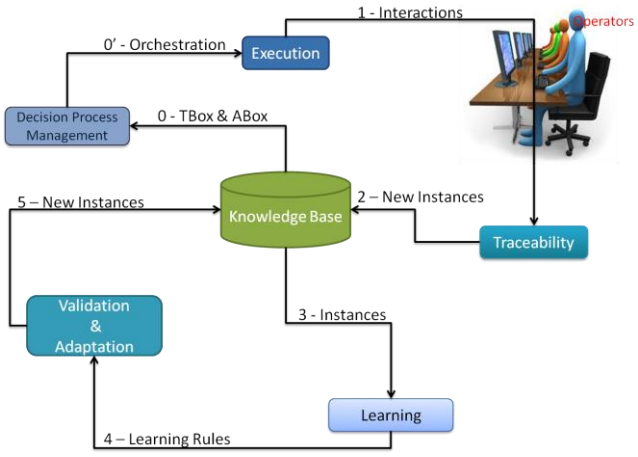

Fig. 4. Closed loop steps for TBS functioning in PHM platform

- **Step 0:** the management module orchestrates the execution of operations of the Decision Support Module by using the knowledge concerning the decision processes (labels 0 and 0 ').
- **Step 1:** Upon execution, maintenance operators interact with the PHM platform to make decisions. A module of "traceability" collects traces' interactions and saves them as new instances in the knowledge base according to a trace model (labels 1 and 2).
- **Step 2:** A learning module retrieves instances concerning traces of the decision process to analyze them and extract new knowledge about these processes (label 3 and 4).
- **Step 3:** A validation and adaptation module related to the analysis module fits the results of the analysis module to respect the persistency of the knowledge base before saving it as new instance in the base (label 4 and 5).

3. TRACEABILITY SERVICE

The traceability service collects all traces recorded using a computer environment safeguarding the activities of maintenance decision makers. This service refers to the collections system in the TBS architecture. The registration of these traces can refer to different formats such as log files, tracks or trails. According to Bousbia, data collection provides "raw traces" or "primitive signs" difficult to use as such (Bousbia 2011).

To exploit the reuse of interaction's experience, it is necessary to model the traces (Cran et al 2007). In this context, Settouti et al formally define a trace as "a collection of observed that can be temporally located." Observed can be any element of the user environment (an entity, an action) that makes sense in the observation of its activity (Settouti et al., 2007).

Every trace observed is an instance of an observed class, which belongs to a hierarchical structure of classes called the model observed trace. Also, it contains the relationships between these classes. Consequently, the observed traces are interconnected via relations instances of relations defined in the model trace.

3.1 Trace model

Therefore, in the case of the decision support module, the traceability service will focus on the level of the users' activities to make decisions. Thus, the trace model to be used should cover the all activities of the maintenance process managed by the users of the PHM platform.

As shown in Figure 5, the model traces can be summarized in the concepts *Process, Activity, Actor and ActivityInPutOutPut* and relations that structure.

A process is a sequence of an interdependent and linked activities *(Acitivity)* in which each activity consumes one or more resources to convert inputs to outputs *(ActivityInPutOutPut)*. These inputs/outputs reference to transitions to the next activity until a known goal is reached (the end of the process).

Instances of the concepts Process and Activity presents the interactions made by the user in the PHM platform and traced by the traceability service. In addition, each activity is performed by one or more actors (users of the platform) that are defined by instances of the concept of "Actor".

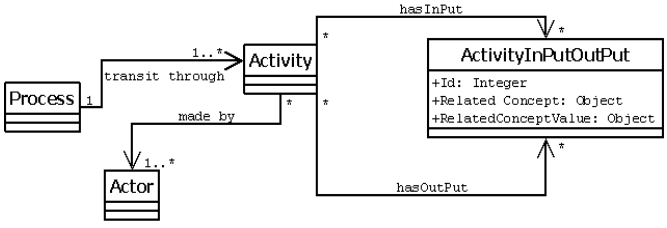

Fig. 5. Trace model for Decision Support Module in PHM platform

3.2 Traceability service

As already mentioned that the log files are a good tool to save the traces, but they are constrained by the complexity of their structure and analysis. Consequently, we used both the log files and modeled traces in the traceability services that we present in Fig.6. So there is a transformation from log file traces to the modeled traces then these latter are stored in the knowledge base. More information about the internal functioning of these activities and their set up are presented in (Karray et al).

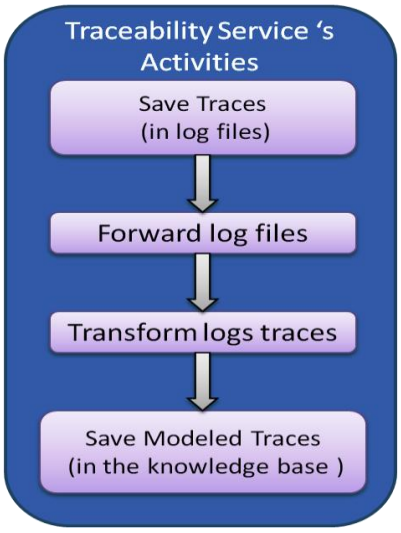

Fig. 6. Traceability service's activities

4. SELF-LEARNING

Once the traces were structured, the learning step allows the interpretation of these traces to extract results and conclusions based on the observation aims (Bousbia 2011). Self-learning service is the service that automates the learning module in step 2 of the closed loop. Also, this service corresponds to the transformation system in the TBS architecture. According to Nilsson, learning, such as intelligence, covers a wide range of processes that are difficult to define precisely (Nilson 1998). Machine learning refers to a system that can automatically acquire and integrate knowledge. The ability of systems to learn from the experience, training, analytical observation, and other means allows to the system to constantly improve itself and have more efficiency and effectiveness.

A machine learning system usually begins with some knowledge and organization of relevant knowledge to be able to interpret, analyze and test the knowledge (Nilson 1998). Develop machine-learning systems without human intervention means to develop self-learning system. Thus, as long as the machine learning techniques are considered the core of any process of self-learning system, this latter is characterized by the self-adjustment of its behavior based on the learned knowledge.

In addition it will be possible for the self-learning service to understand the different relationships and interactions to conclude certain dependencies for generating new decision process models. Then, this new knowledge will be exploited in the process of self-management to adjust the behavior of the decision support module.

Although, even after pretreatment and structuring into the traceability service, the collected traces are generally voluminous and sometimes not expressive. Thus, the self-learning service (associated to analysis phase) processes the traces using generally statistical and/or data mining methods.

To implement this process of self-learning, from different machine learning techniques (Nilson 1998) we adopt the decision tree approach given the ease of manipulating "symbolic" data as well as different amplitude variables and the effective possibilities of classification. According to Nilsson, a decision tree is a tree which its internal nodes are tests (the input model) and its leaves are the nodes types (models). A decision tree assigns a class number (or output) to a model by filtering the input through the model tests in the tree. It should be noted that there are different tools that are transforming the decision tree to decision rules.

Several learning systems of decision trees have been proposed (Nilson 1998). From these different systems we adopt the algorithm C4.5 while it is based on entropy measure in the training data to produce the induction graph. The advantage of using entropy is that the algorithm works on symbolic data that are categorical variables or discreet numerical. Therefore, this algorithm seems the best suited to our needs that we consider as a classification problem.

Xu et al (Xu et al 1995) note that a learning service must satisfy the condition of consistency which means that the learning results must be consistent with the original existing knowledge in the knowledge base. In fact, the self-learning service is mainly based on knowledge models of the knowledge base in the PHM platform.

The launching of the Self-learning service depends to the number of treated RULs (process concerning decision support to treat received RULs) according to a defined threshold. This choice is argued by the learning result depends mainly on the number of instances contained in the dataset to be analyzed by the C4.5 algorithm.

As shown in Fig.7, the first two steps in this process deals with the creation of data views that contain the knowledge needed to analyze from the knowledge base. This step is performed to present the knowledge in a structured form (e.g. table with specific schema).

Then, from these views, the various files needed for the algorithm C4.5 are automatically created. The fifth step is to run the algorithm C4.5.

According to the percentages of correct classification and the percentages of error the system validates or not learning results. The validation is done according to strict thresholds previously defined about these indicators as shown in Table 1. Finally, if validated, results of this learning are transmitted to the component that manages the process of exploitation. More information about the internal functioning of these activities as well as information about the used metrics and thresholds used to validate the learning results are available in (Karray et al 2011).

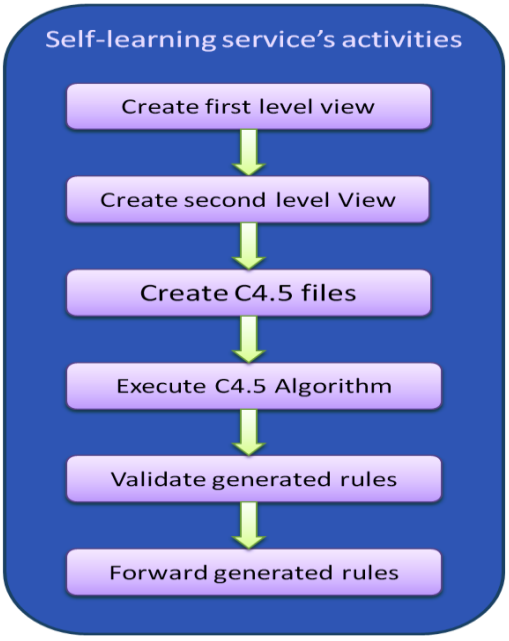

Fig. 7. Self-learning service's activities

Table 1. Validation indicators' thresholds

Indicator	Threshold
Correctly Classified Instances	> = 90%
Kappa statistic	> = 0.5
Relative absolute error	< = 10%
relative squared error	< 50%

5. SELF-MANAGEMENT

The third stage of the closed loop of SBT in the PHM platform is designed to adapt the results of the analysis module and save them as new instances in the knowledge base.

The self-management service is the process that uses this knowledge to automatically manage decision activities on which the learning service has allowed to deduct the way of their implementation and their inputs and outputs values.

So, after adding the knowledge about a given activity in the knowledge base, in the next executions of the PHM platform, this activity will be performed automatically without any intervention by maintenance operators.

Fig. 8 illustrates operations of the self-management service. Indeed, after receiving the learning rules, the service adapts these rules to respect the structure of the knowledge base. Then, the knowledge base is updated with the new rules. We note that this update may take different ways: (1) add new rules and keep the existed rules, (2) add new rules and modify or delete existing rules, and (3) change existing rules. Detailed information about self-management service's functioning is also presented in (Karray et al 2011).

Thereby, the first three activities of this service deal with the third step of the closed loop of TBS functioning. The last activity concerns *Step 0* of the closed loop.

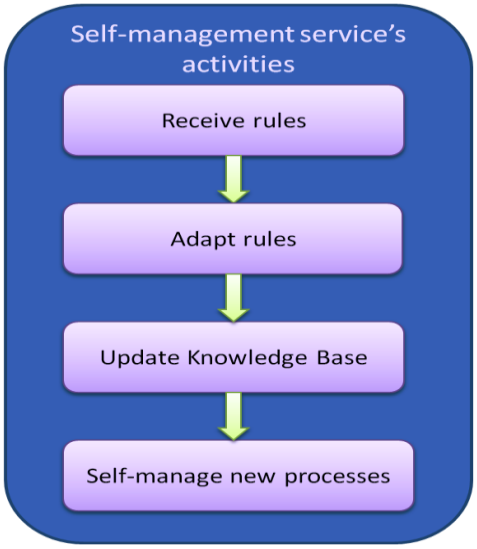

Fig. 8. Self-management service's activities

6. USE CASE

KEM is a company that provides maintenance services for elevators. KEM uses PHM platform to manage 1000 elevators dispatched all over on France. KEM's slogan is "Quality, Speed, Proximity and Reliability". To be in agreement with this slogan, the company is not limited on preventive maintenance (systematic) but it provides also a predictive maintenance service on all the elevators operated by its PHM platform. Hence, sensors are installed on two critical components of each elevator which are pusher and conveyor belt.

KEM estimates that over one year there is two events triggered by two RUL for each elevator which gives an average of 2000 RUL/year besides systematic inspections. RULs are triggered in different contexts. As shown in F1, the decision module provides recommendations to develop an operational scenario according to the *3-tuple (Component, RUL, context)*. The recommended scenario is then executed by maintenance operators via the PHM platform. The different scenarios used by KEM are presented in Table 2.

F1: Decision_Module (RUL, Corposant, context) => operational scenario.

We notice that scenarios in this table may be updated (add, modify, delete) by KEM's maintenance experts.

On the other hand, among these 2000 executions of scenarios related to the received RULs, scenarios' distribution is not necessarily homogeneous. For example, it can be 1500 executions of the first scenario, 300 executions of the third scenario and 200 of the forth. Indeed, the analysis done by the self-learning service is according to the *4-tuple (RUL, Component, context, Scenario).*

As mentioned above, the launching of the self-learning service depends to a threshold about the number of the 4-tuple concerning a particular scenario in the knowledge base. In the scope of KME, maintenance experts define the threshold as 500 registered 4-tuple. So every new 500 4-tuple traced about a particular scenario the service is launched.

Table 2. Different scenarios in the decision module

	Scenario Id	Operational Scenarios
RUL + Context	1	Validation + Scheduling of an Express Intervention
	2	Validation + Scheduling Intervention with normal priority (before the end of the RUL duration)
	3	Validation + Reconfiguration (Reduce the maximum weight by 50%) + Scheduling of an Express Intervention
	4	Validation + Security measures (Secure the elevator [declared as out of service while the elevator is vacuum])+ Scheduling of an Express Intervention
	5	Validation + Reconfiguration (Reduce the maximum weight by 50%) + Assign a notification to the next planned preventive intervention (before the end of the term of the RUL)

Over a period of five months traceability service in the PHM platform of KME records 1105 scenarios including 500 concerning scenario 3. The self-learning service is launched, but the service does not validate the analysis results. After 4 months the number of 4-tuple concerning scenario 3 increases by 500, so the self-learning service stimulus again and this time the service validates a new rule *R1* about the component pusher.

R1: for each pusher in the context Y (elevator type ZE32, sensor measure between 50 and 75) with a defined RUL between 5 and 10 days, scenario 3 is applied.

Therefore, in future executions, thanks to *R1*, when the decision module receives a RUL about a pusher in the context Y the self-management service automatically apply the third scenario by an automatic validation of the RUL, automatic reconfiguration of the maximum weight allowed by the elevator and automatic planning of an intervention before the deadline of the RUL without any operation by the platform user.

The following algorithm summarizes the execution progress of the decision module in the scope of the presented use case:

While execution
  If (RUL_i + Component X+ $context_j$)**not_Included_In** *{Rn}) then*
    (RUL_i + Component X+ $context_j$) => *Decision Process*
    $Actor_k$*(Decision Process)*
    Traceability Service ()
    If ($KB_T = KB_{T-1}+500$) *then*
      Self-learning Service () => New Rule (Rm)
    End if
  else
    RUL_i + Component X+ $context_j$ => *Self-management Service* (S_k)
  End if

End while;

{Rn}: set of rules / S_k*: scenario associated to* R_k

7. CONCLUSION

We proposed in this work a Trace Based System into the Decision Support Module in the scope of a PHM platform. The proposed system provides three main services which are treatability, self-learning and self-management. The proposed services aim to evolve the performances of the platform. Also, we illustrated the functioning of this system by providing a use case concerning the management of elevators by the PHM platform. Concerning future works, we aim evaluate the performance of the SBT in large scale platform and develop a method for self-adjustment of the thresholds conditioning the launch of the self-learning service.

REFERENCES

Xu, C.S.; Xu, Z.M.; Xiao, P.D.; Zhou, Z.Y.; Liu, S.X.; Jiang, Z.H. (1995). A self-learning system and its application in fault diagnosis. Proceedings the Instrumentation and Measurement Technology Conference.

Nilsson, J. (1998). Introduction to Machine Learning: An Early Draft of a Proposed Textbook.. http://robotics.stanford.edu/people/nilsson/mlbook.html. Pages 175-188.

Thurston, M. G. (2001). An Open Standard for Web-Based Condition-Based. Proceedings of AUTOTESTCON, (pp. 401- 415). USA, Valley Forge.

Champin, P.A., Prie, Y., Mille, A. (2004). Musette : a framework for knowledge capture from experience. Extraction et Gestion des Connaissances EGC'04. Clermont Ferrand.

Settouti, L.S., Prié, Y., Mille, A., Marty, J.C. (2006). Système à base de traces pour l'apprentissage humain. TICE, Colloque International en "Technologie de l'Information et de la Communication dans l'Enseignement Supérieur et l'Entreprise". Toulouse, France.

Cram, D., Jouvin, D., Mille, A. (2007). Visualisation interactive de traces et réflexivité : application à l'EIAH collaboratif synchrone eMédiathèque. Journale des Sciences et Technologies de l'Information et de la Communication pour l'Éducation et la Formation Numéro spécial « Analyses des traces d'utilisation dans les EIAH ».

Lee, J., Ghaffari, M., Elmeligy, S. (2011). Self-maintenance and engineering immune systems: Towards smarter machines and manufacturing systems. Annual Reviews in Control Volume: In Press,, Issue: 1, Publisher: Elsevier Ltd, Pages: 111-122.

Karray, M.H., chebel-morello, B., Zerhouni, N. (2011). Self-Management Process in S-Maintenance Platform. Proceedings of the 6th World Congress of Engineering Asset Management, WCEAM'11.Cincinatti USA.

Bousbia, N. (2011). Analyse des traces de navigation des apprenants dans un environnement de formation dans une perspective de détection automatique des styles d'apprentissage. Phd Thesis, Université de Pierre et Marie Curie.